

Mike Vainio

WDS Server for Hybrid Delivery of Windows and Linux Operating Systems

Helsinki Metropolia University of Applied Sciences

Engineer

Communication Networks and Applications

Bachelor's Thesis

21 Nov 2017

Author(s) Title Number of Pages Date	Mike Vainio WDS Server for Hybrid Delivery of Windows and Linux Operating Systems 37 pages + 1 appendixes 21 Nov 2017
Degree	Engineer
Degree Programme	Information Technology
Specialisation option	Communication Networks and Applications
Instructor(s)	Tapio Wikström, Senior Lecturer
<p>The objective of the thesis was to build a standalone server capable of delivering boot media to client computers using the Preboot Execution Environment protocol. The server was delivered for the case company Varian Medical Systems Finland to increase the efficiency of the system administrators.</p> <p>Microsoft Windows was the primary operating system used in the case company's environment thus the Windows Deployment Services role was used to deliver the boot medias over the network. In addition to the Windows image a variety of Linux kernel based tools were added for troubleshooting and recovering systems. To deliver the Linux systems, the PXELINUX boot loader from the Syslinux Project was added to the Windows Deployment Services.</p> <p>The implementation provides clearly documented steps for deploying the solution and includes a PowerShell script for automating most of the workflow of updating the solution for the use of system administrators. The implementation was thoroughly tested and verified that it matches the specifications and is reliable. Network traffic was inspected thoroughly, and the ports and protocols used by the solution were documented.</p> <p>The results of the study show that the solution matches the specifications and needs of the case company and future improvements are suggested for the case company to implement in the future. The solution saves time in the installation phase and enhances the IT support provided by the system administrators by harnessing the potential of the network.</p>	
Keywords	WDS, PXE, Windows, Deployment, Linux

Tekijä(t) Otsikko	Mike Vainio WDS Server for Hybrid Delivery of Windows and Linux Operating Systems
Sivumäärä Päiväys	37 pages + 1 liitettä 21.11.2017
Tutkinto	insinööri (AMK)
Tutkinto-ohjelma	tieto- ja viestintäteknikka
Ammatillinen pääaine	Communication Networks and Applications
Ohjaajat	Lehtori, Tapio Wikström
<p>Insinööriyön tavoitteena oli palvelin järjestelmä joka toimittaa asennus- ja käynnistysmedi- oita tietokoneille tietoverkon yli käyttäen PXE protokollaa. Palvelimen tuotettiin asiakasyri- tykselle Varian Medical Systems Finlandille tehostaakseen yrityksen järjestelmäasiantunti- joiden tehokkuutta.</p> <p>Microsoft Windows on asiakasyrityksessä pääasiallisesti käytetty käyttöjärjestelmä ja tämän takia Windows Deployment Services roolia käytettiin toimittamaan käynnistysmedia tieto- verkon yli. Windows median lisäksi järjestelmään lisättiin myös Linux-kerneliin perustuvia erilaisia työkaluja järjestelmien vian etsintään ja tiedon palauttamiseen. Näiden Linux-poh- jaisten työkalujen toimittamisen mahdollistamiseksi Windows Deployment Services -palve- luun lisättiin Syslinux projektista PXELINUX alkulatausohjelma.</p> <p>Opinnäytetyö tarjoaa selkeästi dokumentoidut ohjeet ratkaisun toteuttamiseen ja myös Po- werShell ohjelman ratkaisun päivitysprosessin automatisointiin asiakasyrityksen järjestel- mäasiantuntijoiden käyttöön. Ratkaisun toiminta ja luotettavuus testattiin sekä varmistettiin vastaavan suunnitelmaa. Tietoverkon liikennettä tarkisteltiin ja järjestelmän käyttävät portit sekä protokollat dokumentoitiin.</p> <p>Työn lopputuloksen todettiin vastaavaan asiakasyrityksen tarpeita ja suunniteltua kokonai- suutta, myös tulevaisuuden varalle ehdotettiin parannuksia sekä muutoksia ratkaisuun. Luotu järjestelmä säästää aikaa tietokoneiden käyttöönotossa ja tehostaa asiakasyrityksen tietohallinnon tukea tuotannolle käyttämällä tietoverkkoa apuna.</p>	
Keywords	WDS, PXE, Windows, Käyttöönotto, Asennus, Linux

Contents

1	Introduction	1
1.1	Company background	1
1.2	Research problem and objective	2
1.3	Structure of thesis	2
2	Project Specifications	2
2.1	Current state	2
2.2	Project design	3
2.3	Comparing solutions	3
2.3.1	Windows Deployment Services	3
2.3.2	FOG	4
2.3.3	Serva	5
3	Theory	7
3.1	Preboot Execution Environment	7
3.1.1	PXE boot process	7
3.2	Network requirements	8
3.3	UEFI versus BIOS	8
3.4	History of WDS	9
3.5	How WDS works	10
3.6	Linux	10
3.6.1	Deploying Linux	11
3.7	Backup and recovery tools	11
4	Implementation	13
4.1	Architecture	13
4.2	Installing the WDS role	14
4.2.1	Initial configuration of the WDS role	16
4.3	Preparing the SCCM image to be used in WDS	17
4.4	Adding boot image to WDS	19
4.5	Deploying Linux operating system	22
4.5.1	Ubuntu Live CD	23
4.6	Adding backup and recovery tools	27
4.6.1	Memtest86+	27
4.6.2	System Rescue CD	28

4.7	Automating the boot image upload	30
5	Testing and verification of solution	32
5.1	Network	32
5.1.1	UEFI PXE boot	32
5.2	Comparing UEFI and BIOS/Legacy PXE boot	34
5.3	Hardware compatibility	34
6	Security	35
6.1	Securing PXELINUX with a passphrase	35
7	Conclusions	36
7.1	Future improvements and ideas	36
	References	38
	Appendices	
	Appendix 1. PowerShell script for updating boot image	

ABBREVIATIONS

DHCP – Dynamic Host Configuration Protocol

PXE – Preboot Execution Environment

NBP – Network Bootstrap Program

TFTP – Trivial File Transport Protocol

UEFI – Universal Extensible Firmware Interface

BIOS - Basic Input Output System

GPT – GUID Partition Table

FAT32 – File Allocation Table 32-bits

NTFS – New Technology File System

MBR – Master Boot Record

WDS – Windows Deployment Services

SCCM – System Center Configuration Manager

MDT – Microsoft Deployment Toolkit

NIC – Network Interface Card

RHEL – Red Hat Enterprise Linux

RAM – Random-Access Memory

VM – Virtual Machine

AD DS – Active Directory Domain Services

WIM – Windows Imaging File Format

ISO – ISO image file

ADK – Assessment and Deployment Kit

NFS – Network File System

LAN – Local Area Network

GUI – Graphical User Interface

1 Introduction

Installing the operating system is usually the first step in any company when a computer arrives to the premises. To standardize the configuration and prevent human errors during the deployment of an operating system on the computers is often automated to a certain degree. Many tools can be used for automating the operating system deployment and some of the most common ones used in the industry are explained in the theory chapter of this thesis.

Deploying the operating system to multiple computers concurrently frees up resources to the supporting IT team. This can be accomplished by delivering the installation media with the network. Utilizing the network to load bootable images instead of a removable media is more convenient as the deployment of any client operating system requires the network connectivity for downloading the critical security updates before the system can be handed over to the end-user.

System administrators do not have to carry and update removable media for deploying or troubleshooting a device as the solution provides a wide variety of troubleshooting and recovery tools. The solution presented in this study saves time and thus saves money for the case company, it is also highly versatile and cost-efficient to implement in most environments.

1.1 Company Background

Varian Medical Systems produces tools that harness the power of X-ray energy for the benefit of humankind. Varian produces devices and related software in the fields of radiotherapy, radiosurgery-ray tube technology and many others. Varian has over 70 offices around the world and over 6600 people working for the company. Varian's branch office in Finland employs over 160 people and focuses on software development. Employees in the branch office collaborate with Varian employees all over the globe.

1.2 Research problem and objective

System administrators of the case company currently use removable media such as CD/DVD and USB storage to store and deliver the boot media when deploying new machines or when backup and troubleshooting tools are needed. Using removable media has many security concerns and it does not scale for high volume deployment. Keeping track of the different medias and their location while keeping the media up-to-date is frustrating and wasting time as all the machines are connected to the network which could be utilized to deliver the boot media. To tackle this problem, the objective of this study is to implement a standalone system for delivering hybrid boot media utilizing the network and thus rendering the removable devices obsolete.

1.3 Structure of thesis

This study has seven sections. Section 2 provides the specifications set for the solution and a brief comparison of other similar solutions available. Section 3 explains the theory behind the solution and introduces all the building blocks of Section 4, which presents the implementation of the solution. Section 5 verifies that the solution functions as specified in Section 2 and that the Section 3 theory is applicable, while Section 6 focuses on securing the solution from threats and Section 7 concludes the study with the final words.

2 Project Specifications

2.1 Current state

Currently the deployment of new PC's is done by using removable media, such as a USB stick or a DVD. The system saved in the removable media is a bootable image exported from the SCCM server, which contains a WinPE operating system capable of initiating the task sequence, which will perform the installation of the Windows operating system. The task sequence is password protected and only the system administrators know the passphrase. The usage of this bootable media is allowed only for people working in the IT department. The task sequence for production is created by a global team and this process is out of the scope of this study.

This study does not support the deployment of Server operating systems because of the low volume of installations. Instead the primary focus is on installing the Windows 10 operating system and delivering troubleshooting tools reliably, while saving time.

2.2 Project design

This project leans on the existing infrastructure, but still delivers a completely standalone and locally managed solution to load bootable operating system images for the Helsinki branch office. The server will be hosted with the same requirements and security in place as with other core infrastructure servers such as a domain controller and a DHCP server.

In addition to the boot image for the Windows operating system deployment this project also focuses on delivering bootable Linux distributions and recovery and diagnostics tools over the network. The project starts with first implementing the deployment of the case company's production image. After this, the work with the Linux operating system, backup and diagnostic tools will be implemented. Once all the work has been conducted a private testing will be conducted by the IT department and after that the solution is tested by production use. Primarily, the solution is focused on making the IT support more effective and saving time. Linux and the other tools can be used also by the R&D engineers and they can start using the system after the testing is done by the IT department.

2.3 Comparing solutions

In this section, three popular and effective imaging solutions are introduced and compared with each other. The solutions compared in this study are FOG, Serva and Microsoft WDS all of which are briefly introduced.

2.3.1 Windows Deployment Services

Windows Deployment Services (WDS) is a Windows server role developed by Microsoft. WDS delivers boot image utilizing the network instead of DVD's or USB storage devices for the initial first boot. WDS can be used as a standalone or it can further append Mi-

Microsoft System Center Manager (SCCM) or Microsoft Deployment Toolkit (MDT) deployment features. WDS supports all popular architectures, including ARM and 32/64-bit UEFI boot images. WDS will be later described in detail in Section 3.

2.3.2 FOG

FOG is an open source cloning solution and it is widely used by many organizations such as schools and other non-commercial institutions. FOG is installed on a computer or virtual machine (VM) running Linux such as Ubuntu or CentOS. The FOG server is managed then by using a web user interface (UI). FOG supports the imaging of Windows operating systems (excluding Windows Server), Linux and MAC OS X. FOG also offers computer management features for installing software and running scripts on remote client computers. FOG offers a great number of features especially for being a free solution, but looking at the wiki and forum of the project many hardware related issues are reported. Especially Dell and HP desktops, laptops and servers seem to have known problems. FOG is complicated to setup comparing to WDS for example and must be updated manually. [1]

2.3.3 Serva

Serva is an Automated PXE Server Solution Accelerator according to the official website and offers the following services:

- HTTP server
- FTP server
- TFTP server
- TFTP client
- DHCP server
- proxyDHCP server
- BINL server
- DNS server
- SNTP server
- SYSLOG server

Serva differs from the traditional PXE server solutions quite dramatically by being a portable application, which is only an approximately 3 MB sized executable file. Serva runs on any Windows version from Windows 2000 to Windows 10, unlike many open source alternatives that run on Linux systems. As the portable nature of Serva implies it does not require installation and is thus very fast to implement. Configuring Serva is a very simplified task when comparing to WDS. Configuration settings are illustrated in figure 1.

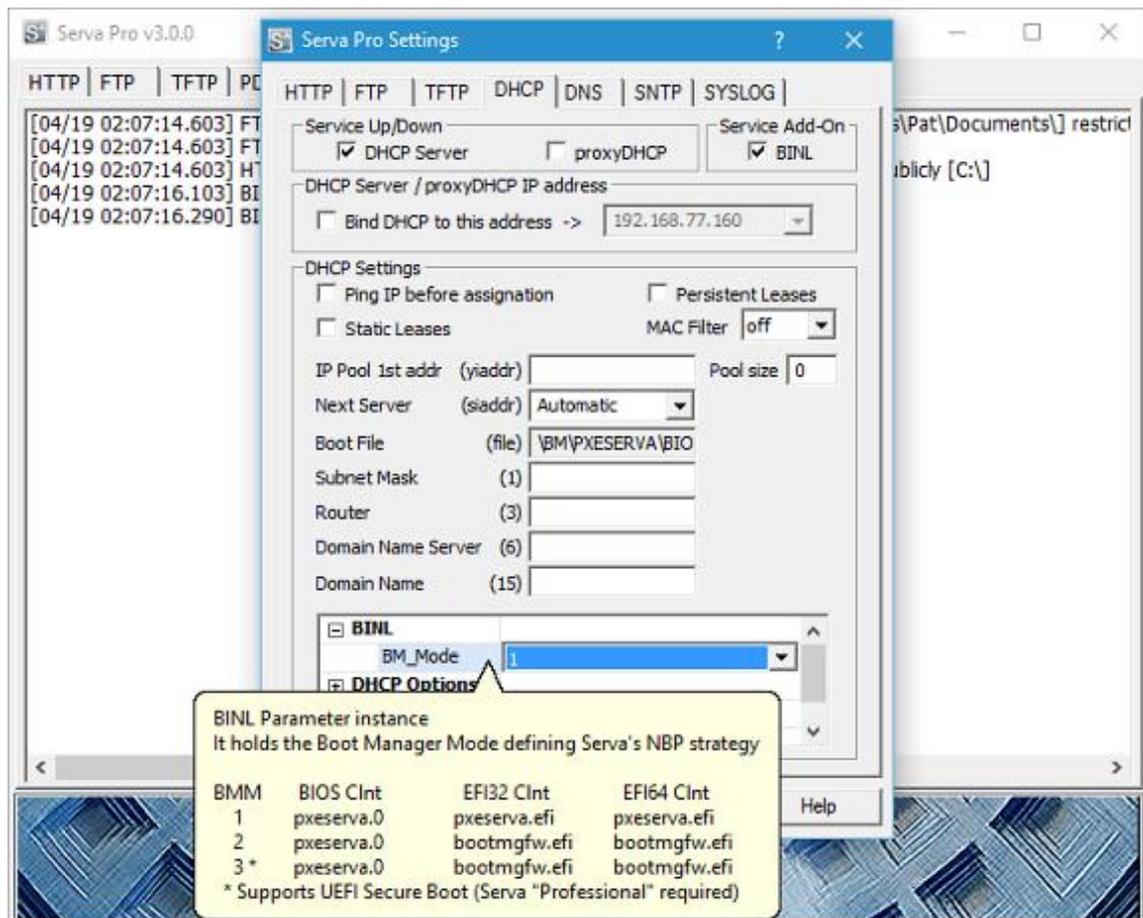


Figure 1: Serva Settings [2]

As figure 1 shows, Serva is configured using just a single menu. However, Serva is not open source and it is also not free for commercial use. Serva's "Community" version, which is free of charge for personal use also lacks several of the features of the professional version. Serva offers a professional license for both 32-bit and 64-bit architectures for a price of €67,50 for a single instance at the time of writing this thesis. The official documentation available at the time of writing this thesis is quite short and clearly not written for complex environments as the deployment scenarios are quite simple. Considering the lacking documentation, the case company of the project would most likely hesitate to buy the license. [2, 3]

Windows Deployment Services was chosen over these alternatives because it is made by Microsoft for the pure intent of deploying Microsoft Windows operating systems and the focus of this study is the Windows deployment. It requires a Windows Server license and has no further costs.

3 Theory

This chapter introduces and explains briefly the key systems used in the solution presented in the next chapter, which concentrates on the implementation. It is assumed in this section that the reader has a basic understanding of information networks, Linux and Windows environments.

3.1 Preboot Execution Environment

Preboot Execution Environment (PXE) is a protocol created by Intel. It specifies a standardized environment for network booting client devices. The client only needs a PXE-capable Network Interface Card (NIC) to implement it and for network addressing PXE relies on a pre-existing DHCP server. To boot via the network a Network Bootstrap Program (NBP) needs to be downloaded from the server. For the download Trivial File Transfer Protocol (TFTP) is used. [4]

3.1.1 PXE boot process

The client following the PXE protocol first broadcasts the DHCPDISCOVER frame containing an extension indicating that the request is from a PXE implementing client. The DHCP server answers with a DHCPOFFER that includes the IP-address for the client. Also, the boot server must respond to the client with the NBP file name and IP address of the TFTP server. The client then accepts the DHCP offer by sending out a DHCPREQUEST and DHCP server answers with the DHCPACK. After this, the client continues to download the NBP file with TFTP from the server. The client executes the NBP file once it is downloaded. Figure 2 presents the boot process: [5, 6]

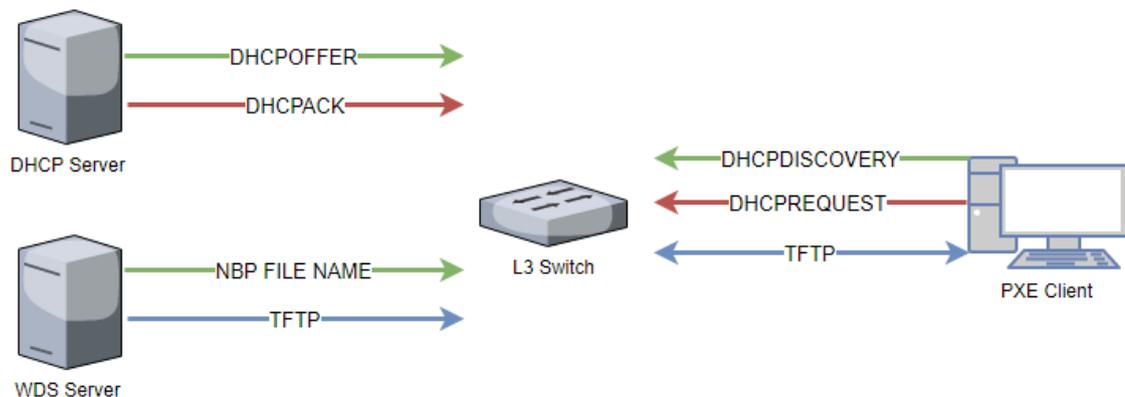


Figure 2: PXE boot process

In figure 2, the L3 Switch has an IP Helper configured with both DHCP and WDS server addresses. The switch forwards the DHCPDISCOVER to both servers, so both servers will receive the packet even when in a different subnet than the client.

3.2 Network requirements

For the PXE boot to work the DHCP server must receive the initial DHCPDISCOVER broadcast from the client. If the client and server are in different broadcast domains a DHCP relay also often called IP helper needs to be configured in the network devices. The IP helper relays the DHCPDISCOVER as unicast to the configured IP-address of the DHCP server, if the DHCP server and the actual server containing the boot file are different, both servers need to be configured to the IP helper. The IP helper forwards the DHCPDISCOVER as unicast to the server(s). If a firewall is located between the client and server, it is specified that the following ports and protocols must be allowed for the server:

- TFTP server UDP port 69
- DHCP server UDP port 67 and 68
- UDP port 4011 for PXE
- Random ports from 64001 through 65000 for TFTP
- TCP port 135 and 5040 for RPC
- TCP ports 137-139 for SMB

Another consideration when using dedicated DHCP and boot servers is that a network device might drop the packets sent by the boot server after the client has accepted the IP-address provided by the DHCP server. This feature is often called DHCP snooping. Usually this is not an issue if the client is only booted once using PXE, but booting it again after a short time would fail because of the DHCP snooping blocking the boot server as a rogue DHCP server. [5, 7]

3.3 UEFI versus BIOS

Unified Extended Firmware Interface (UEFI) is a standardized firmware developed to replace the Basic Input Output System (BIOS). UEFI specifies the interface between an operating system and firmware as illustrated in figure 3:

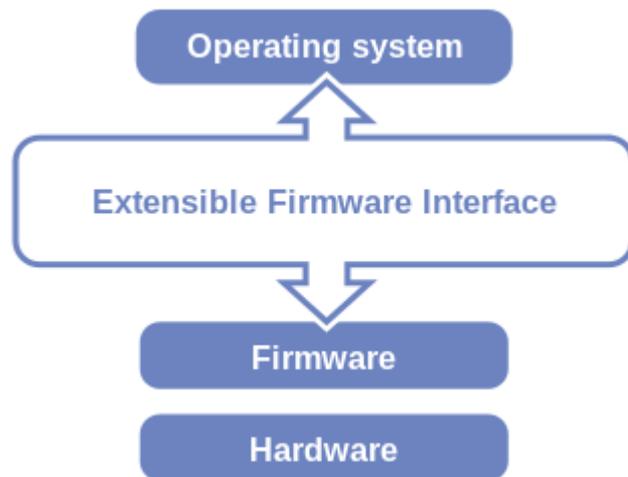


Figure 3: UEFI [8]

As figure 3 presents, the UEFI is a very low-level piece of software. Over 140 technology companies participate in the UEFI forum and all new devices shipped with Windows 10 have an UEFI firmware by default. UEFI provides benefits and removes limitations that the BIOS has. The benefits of UEFI are:

- Faster boot and resume times
- Security features such as Secure Boot and factory encrypted drives
- Support for more than 4 partitions on a single disk and disk over 2 TB
- Support for UEFI drivers and applications
- Backwards compatibility with BIOS known as legacy boot

Before installing the operating system, the choice must be made to use either UEFI or BIOS. If the computer supports legacy BIOS modes the boot media, WinPE, must be started using the correct mode as in UEFI or legacy BIOS. When installing on a drive that has been previously used with BIOS mode, the disk must be changed to use the GPT/FAT32 and NTFS scheme rather than the MBR/NTFS scheme. UEFI does not support cross-platform start up, thus the correct media must be used for 32-bit and 64-bit architecture computers. [9, pp 58-59]

3.4 History of WDS

Microsoft first developed the Remote Installation Services (RIS), which was included in Windows Server 2000 and Windows Server 2003 to enable deployment through the network. RIS was known as difficult to manage and slow. For these reasons, a great number

of organizations rather used a third-party solution for deploying Windows operating systems. Many of these solutions required expensive licensing. Windows Deployment Services (WDS) was introduced in the Windows Server 2003 Service Pack 2 and it has been included in every version of Windows Server operating systems ever since. In Windows Server 2008 came the support for high volume deployments by utilizing multicast for the transfer of image to clients. One of the major benefits of the WDS is that it is included in the Windows Server as a role and requires no additional licensing, unlike many of the third-party options available in the market. [4]

3.5 How WDS works

WDS relies on the PXE technology developed by Intel and included in most of the Network Interface Cards (NIC) of modern computers. For the image file transfer, the Trivial File Transfer Protocol (TFTP) is used. With WDS the client computer can be booted with only the network connectivity and the administrator can choose from a list of boot images available on the WDS server. WDS utilizes two images for the deployment, a boot image and an installation image. In this project only the boot image is used which contains a minimalistic Windows Operating system called WinPE (Windows Preinstallation Environment). The WDS server only provides the boot image, because the boot image used is customized with the Configuration Manager console and includes a task sequence which will perform the operating system install. The SCCM infrastructure could be used to provide a PXE boot service on the distribution point, but that would not be as standalone and versatile as deploying a WDS server. [4]

3.6 Linux

Linux differs from the Microsoft Windows Operating System in many ways. Linux operating systems, usually called distributions, are built around the Linux kernel. A great number of Linux distributions exist and many of them are community-driven open source projects, but also commercial distributions such as Red Hat Enterprise Linux, SUSE and Ubuntu exist which sell support for their products. Due to the vast amount of different distributions, it is not possible to test and verify that all of them work in the solution presented in this project. Rather in this project the focus is to verify that a very popular and well supported distribution Ubuntu can be delivered reliably and effectively.

3.6.1 Deploying Linux

No tool is offered by Ubuntu for an automated deployment of the distribution, but the community has developed open source tools which can deliver the functionality.

Cobbler is a Linux installation server that allows the rapid setup of network installation environments according to the official website. Cobbler builds on the RHEL mechanism Kickstart, which is used to automate the installation of a Linux distribution. It supports the PXE network boot and offers installations in a virtualized environment also. Cobbler can also be used to manage the clients after deployment and it integrates with Yum RPM packet manager. [10, 11]

Unfortunately, it is not possible to implement another standalone system for the Linux deployment and it is also not possible to integrate any of the available solutions with the WDS. Because of this, the WDS system will be modified to include a part of the Syslinux project to provide the interface between Linux and Windows systems. The Syslinux Project is a package of lightweight bootloaders for different scenarios. In this case, the PXELINUX package is used as the bootloader in the WDS environment. [12]

3.7 Backup and recovery tools

Computers are complex systems and both the hardware, and the software will have and have bugs, problems and malfunctions. Firmware bugs can render the hardware unusable or harm the data processed in the computer. Just starting an operating system is a complex matter and requires correct behavior of the hardware and firmware on the motherboard. When things do go wrong, and the operating system will not start, a recovery tool is often needed to fix the system or if the system is beyond saving a backup tool is needed to recover the data. To help make the life of the system administrators easier this study will present how to add useful operating system and data recovery software to the WDS server.

The Ubuntu Linux distribution offers a feature called “Live CD”, this can be used to test the distribution before committing with the installation or as a rescue mode. In live mode, the distribution is downloaded purely to RAM and thus the live mode can run on top of the primary OS such as Windows temporarily. This can be useful if the primary OS will not boot anymore or login credentials are missing. This live CD mode is available by

default in the Linux installation media. The Ubuntu live mode also includes the GUI, unlike actual rescue modes in most distributions. The image used for the Live CD can be customized to include additional programs and other features.

Memtest86+ is probably the most popular and widely used stand-alone, bootable and free software for testing the RAM memory of any computer. Also, a software called Memtest86 exist which is based on the original Memtest86 software just as the Memtest86+ is. According to the Memtest86+ website, “The first version of Memtest86+ was released on early 2004, based on memtest86 v3.0 that was not updated since mid-2002”. [13] The Memtest86 website backs up the story, “During the time period of the MemTest86 V3.0 release (years 2002 to 2004) the code was 'forked' by Samuel Demeulemeester (now part of the French CanardPC publishing group) into a another version of the software called MemTest86+”. [14] Memtest86 is available as “Pro” and “Free” versions and as the naming suggests only the “Free” version is available for free. However, because Memtest86+ is completely free and open source it was chosen for this study. The major feature missing from Memtest86+ is UEFI compatibility, but it is not needed in this study as the hardware supports the legacy BIOS boot.

According to the official website, “SystemRescueCd is a Linux system rescue disk available as a bootable CD-ROM or USB stick for administrating or repairing the system and data after a crash”. [15] System Rescue CD accomplishes these tasks by offering a wide range of system tools for diagnosing and repairing issues. Some of these tools are:

- GNU Parted
- GParted
- FSArchiver
- Ddrescue
- NTfs3g
- Sfdisk
- Rsync

Most of these tools are associated with accessing the data on the hard drive of the broken system and then backing up the data. System Rescue CD also has good documentation on how to use some of the tools to perform certain tasks. Customizing the image is also possible and documented in the official website. [15]

4 Implementation

This part of the study focuses on the implementation of the project according to the project specifications. Installation and configuration of the systems are introduced in the order of which they were implemented to the solution.

4.1 Architecture

For this project, a new dedicated server was required and specified by the client. This server was provided by using the existing VMWare Virtualization Environment. The infrastructure of the virtual environment is a cluster of ESXi hypervisors, controlled by a vCenter and administrated by using vSphere management console. This kind of solution is often referred to as the vSphere platform when using the VMWare terminology. Utilizing virtualization to deploy the server increases the cost efficiency greatly as the WDS server is most of the time only listening for possible clients and requires very minimal CPU and memory resources at that time. When using virtualization, the resources can be freed for other virtual machines.

This dedicated server will run the newest version of Windows Server operating system available, Windows Server 2016. In this server, the WDS role will be installed. A new VM was created with a comfortable amount of resources to run the operating system and the services. The VM was created with 2 virtual hard drives, one with 60 GB of storage for operating system files and one with 100 GB of storage for the boot images.

To enable the WDS server to be reached by client computers, the ESXi cluster is connected to the physical network. A high-level topology of the network is illustrated below in figure 4:

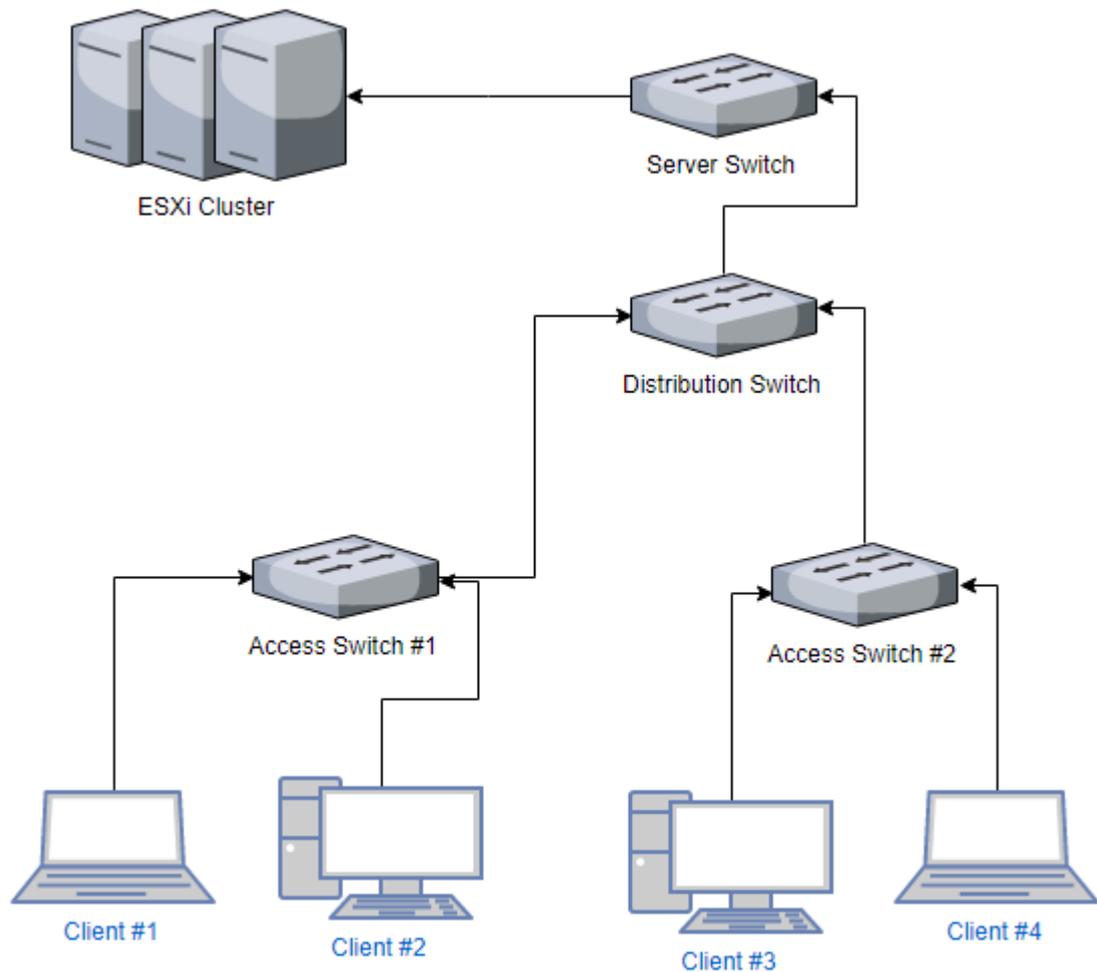


Figure 1: Simplified Network topology

As seen in figure 4, the network consists of multiple subnets and a distribution switch is used to route the traffic between the subnets. In the implementation of this study, the IP helper is configured in every access switch and in the server switch to make sure that every DHCPDISCOVERY broadcast is routed to the DHCP server and WDS server.

4.2 Installing the WDS role

After Windows Server 2016 has been installed and joined to domain with all the required security updates installed, the WDS role can be added. Before starting the installation of the WDS role a static IP address should be configured on the server as the server can only provide the services if it always has the same IP address. It is a good practice to use an IP address that is excluded from the DHCP scope. When installing Active Directory Domain Services (AD DS) role and WDS in the same server the AD DS role should be added first.

To begin installation of the WDS role click “Add roles and features” in Server Manager to open the wizard. Select the local server and go to “Server Roles” page to select the “Windows Deployment Services” check box.

Continue the wizard until the “Role Services” page is reached. It is possible to only install the “Transport Server” service, which provides all the PXE and networking functionality, but does not include the complete functionality to deploy Windows operating systems. The “Deployment Server” services are dependent on the Transport Server. In this study, both services were installed for the full functionality of the WDS role. Figure 5 shows the “Role Services” page:

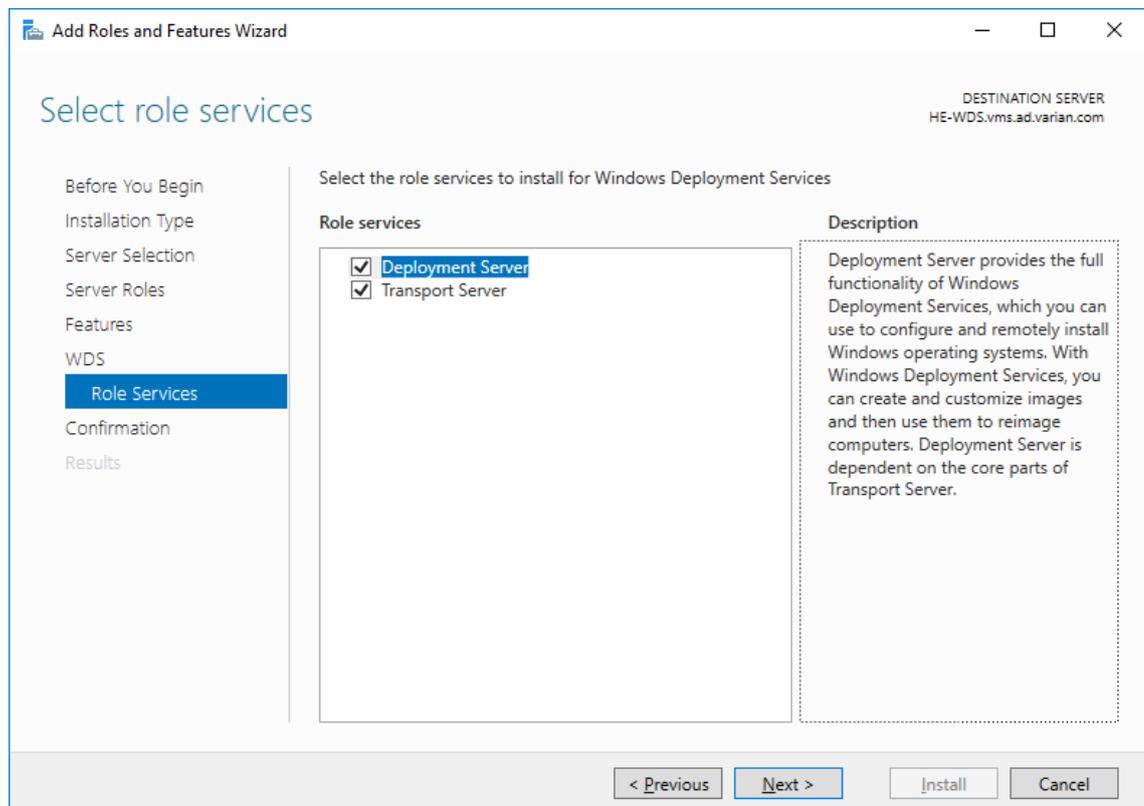


Figure 5: Deployment or Transport

As figure 5 presents, it is possible to only install the “Transport Server” which lacks some of the features used in deploying operating systems. After selecting the role services, continue to the installation and wait for the installation to finish. It is a good practice to restart the server after installing a role and moving on to configuring it. [16]

4.2.1 Initial configuration of the WDS role

Before continuing with the configuration, these requirements must be met:

- The server is part of an AD DS domain or AD DS domain controller to allow the full functionality, or alternatively configure the server in the standalone mode
- DHCP server and a DNS server exist on the network
- The server has a NTFS partition for storing the images

After installing the WDS role open the “Windows Deployment Services” application and select the WDS server for configuration to launch the configuration wizard as seen in figure 6:

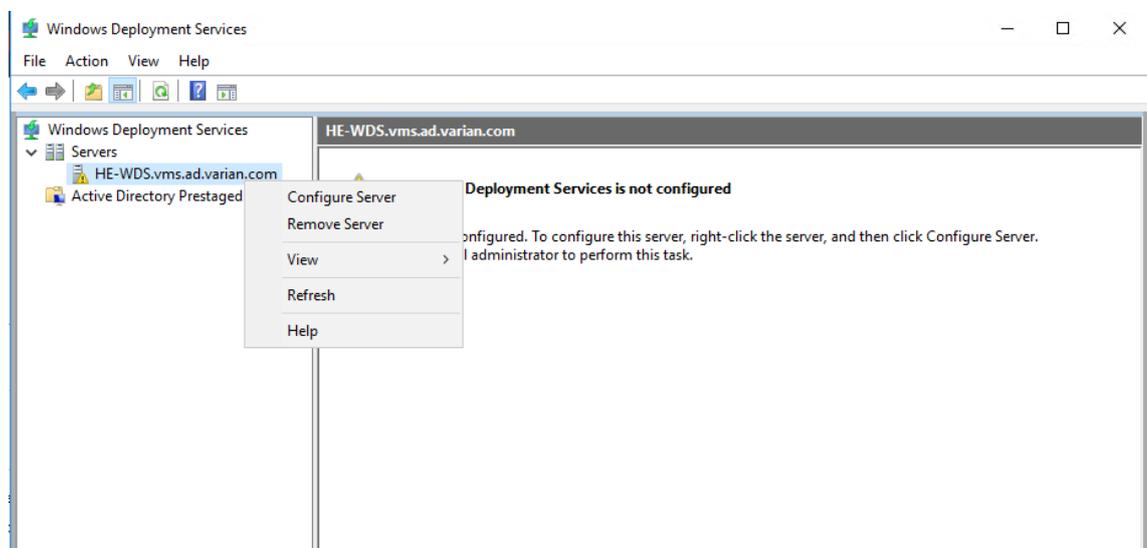


Figure 6: Configure Server

Figure 6 shows that the “Windows Deployment Services” application also indicates that the server is not yet configured. In this project, the server was integrated with Active Directory, but the server can also be run in the standalone mode if it is not part of a domain. The remote installation folder was created on the dedicated hard drive. The next step is to select to which clients the WDS server responds as seen in figure 7:

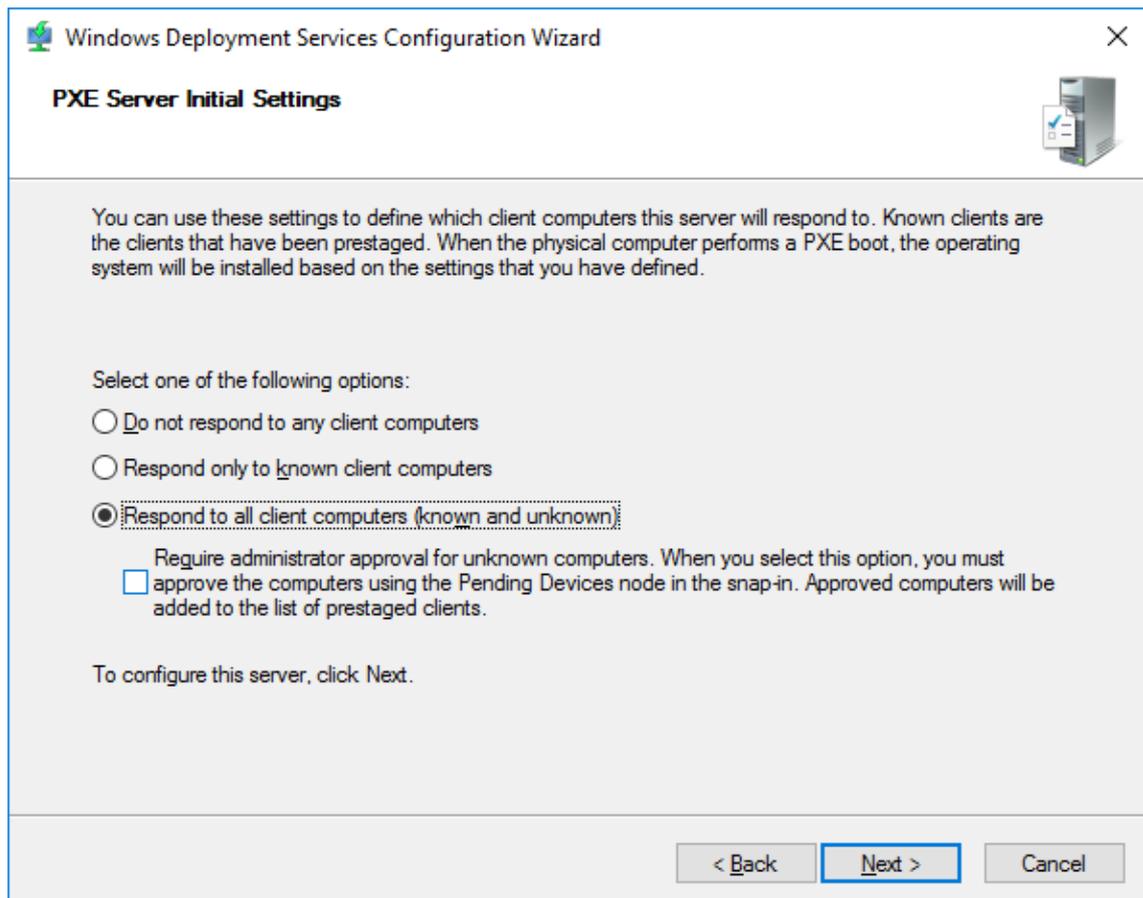


Figure 7: WDS Configuration Wizard

Figure 7 presents all the possible options for defining which client computers the server will respond to. In this project, the WDS server is available to all clients as the other options require manual approval and this would greatly decrease the effectivity. After this selection the WDS role configuration starts and this completes the installation of the role. Next, the WDS server needs a boot image to serve the clients with. [16]

4.3 Preparing the SCCM image to be used in WDS

In SCCM a task sequence media is exported as bootable media. This image can be used as the boot media to start the installation over the network. The image exported is a Disk Image File also called an ISO-file. Only WIM images are compatible with WDS. This ISO contains a compatible boot image in the WIM format, but extracting it requires a few steps from the system administrator.

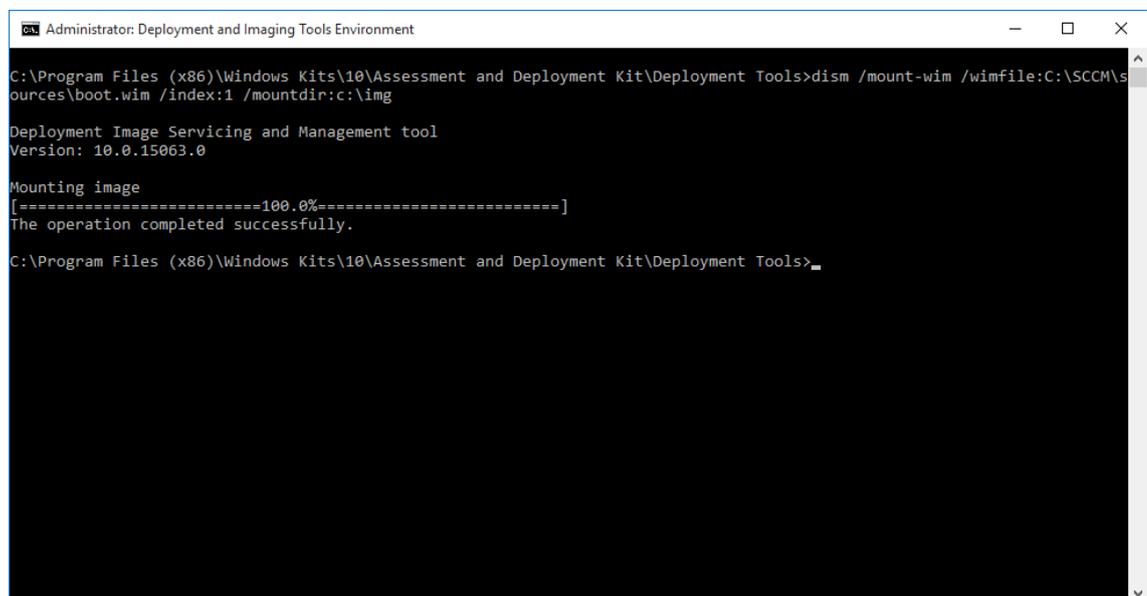
For working with WIM images Microsoft offers Deployment Image Servicing and Management (DISM) tool, which is a part of Windows Assessment and Deployment Kit (Windows ADK). If deploying Windows 7 or older Windows Operating System the Windows

Automated Installation Kit (Windows AIK) is needed. The following steps are performed on a Windows 10 system running Windows ADK version 1703 and might not apply to other versions.

First, the ISO-file is extracted using WinZip or by mounting it and copying the contents to an empty folder. In this case, the ISO file is extracted to “C:\SCCM”. To use DISM the command line must be launched as administrator. Mount the boot file located under the “C:\SCCM” folder to “C:\img” folder with following command:

```
dism /mount-wim /wimfile:C:\SCCM\sources\boot.wim /index:1
/mountdir:c:\img
```

Figure 8 shows the command line interface:



```
Administrator: Deployment and Imaging Tools Environment
C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Deployment Tools>dism /mount-wim /wimfile:C:\SCCM\sources\boot.wim /index:1 /mountdir:c:\img

Deployment Image Servicing and Management tool
Version: 10.0.15063.0

Mounting image
[=====100.0%=====]
The operation completed successfully.

C:\Program Files (x86)\Windows Kits\10\Assessment and Deployment Kit\Deployment Tools>
```

Figure 8: CMD

As seen in figure 8 the image mounted successfully to the folder C:\img.

Copy DATA-folder from “C:\SCCM\SMS” to “C:\img\SMS”. To enable the command line in the WinPE, locate the tsbootshell.ini file in the “C:\img\SMS” folder and add the line:

```
EnableDebugShell=True
```

This is useful to debug problems in deployment. Now the image can be unmounted:

```
dism /unmount-wim /mountdir:C:\img /commit
```

Now the file “C:\SCCM\sources\boot.wim” is ready to be imported to the WDS server.
[17]

4.4 Adding boot image to WDS

Boot images contain the WinPE that the client first boots to. The image uploaded to the server was the case company’s production image prepared in the last section. To begin open the “Windows Deployment Services” application. In the application, expand the “Servers” selection and then expand the appropriate server to browse the images. To add the boot image, use the right click and select “Add Boot Image...” as illustrated in figure 9:

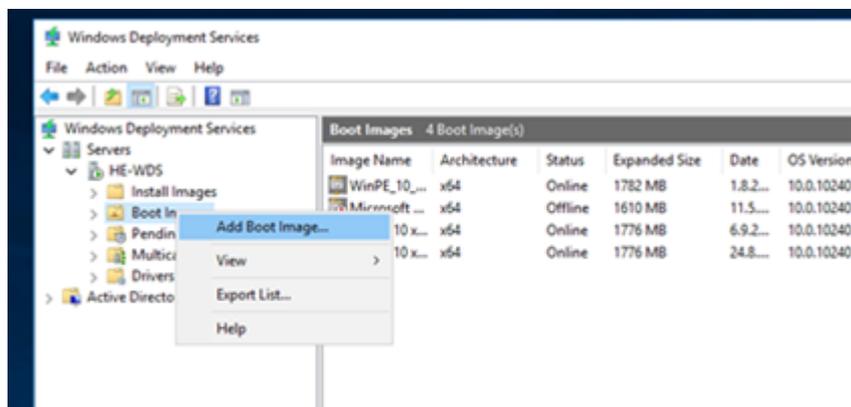


Figure 9: WDS Boot Images

Figure 9 shows the GUI where the boot images are managed. After launching the “Add Image Wizard” the boot image is selected first. In this project, a shared network folder contains all the image files and only the IT-personnel can access this folder and the files. This helped to streamline the process of updating boot images.

Select the applicable image and continue the setup as illustrated in figure 10:

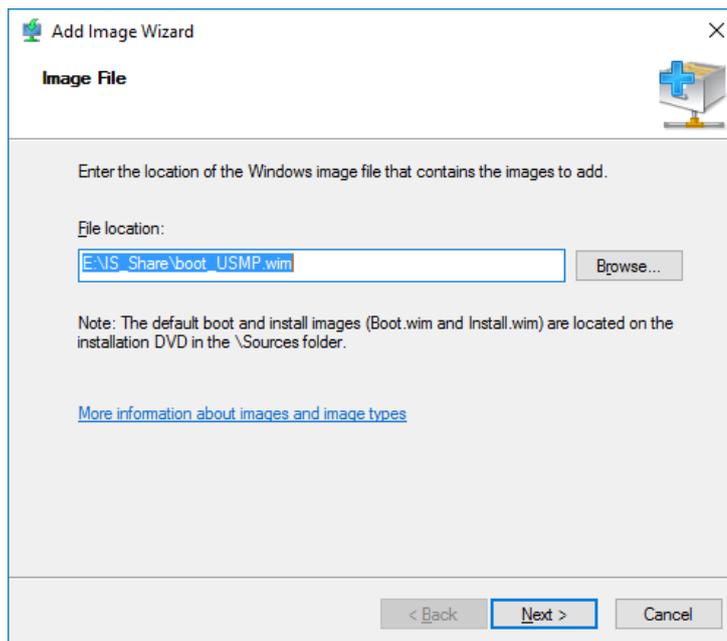


Figure 10: Boot Image uploaded to the network share

Figure 10 shows an example of the image path and type.

After the image is selected, continue the wizard by clicking next. Now the image must be named. The name given here will be visible for the PXE client. If multiple boot images are available, a list of available images is shown to the user, so image names should be descriptive. After this step, the boot image is loaded into the service, but in this case the image is also set as the default boot image. To set the boot image as the default image in WDS service, open the properties for the applicable server as illustrated in figure 11:

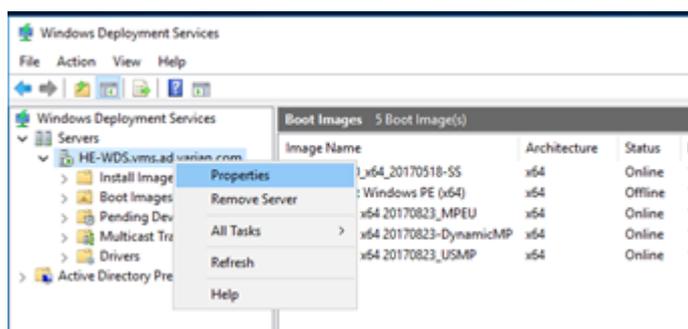


Figure 11: WDS Server Properties

Figure 11 shows that the properties for the server is available after configuring.

In the properties dialog, open the “Boot” tab. In the boot tab, it is possible to select the parameters for the initial boot behavior, but most importantly the “Default boot image”

can be set to allow the automatic boot of the preferred image. Refer to figure 12 for example:

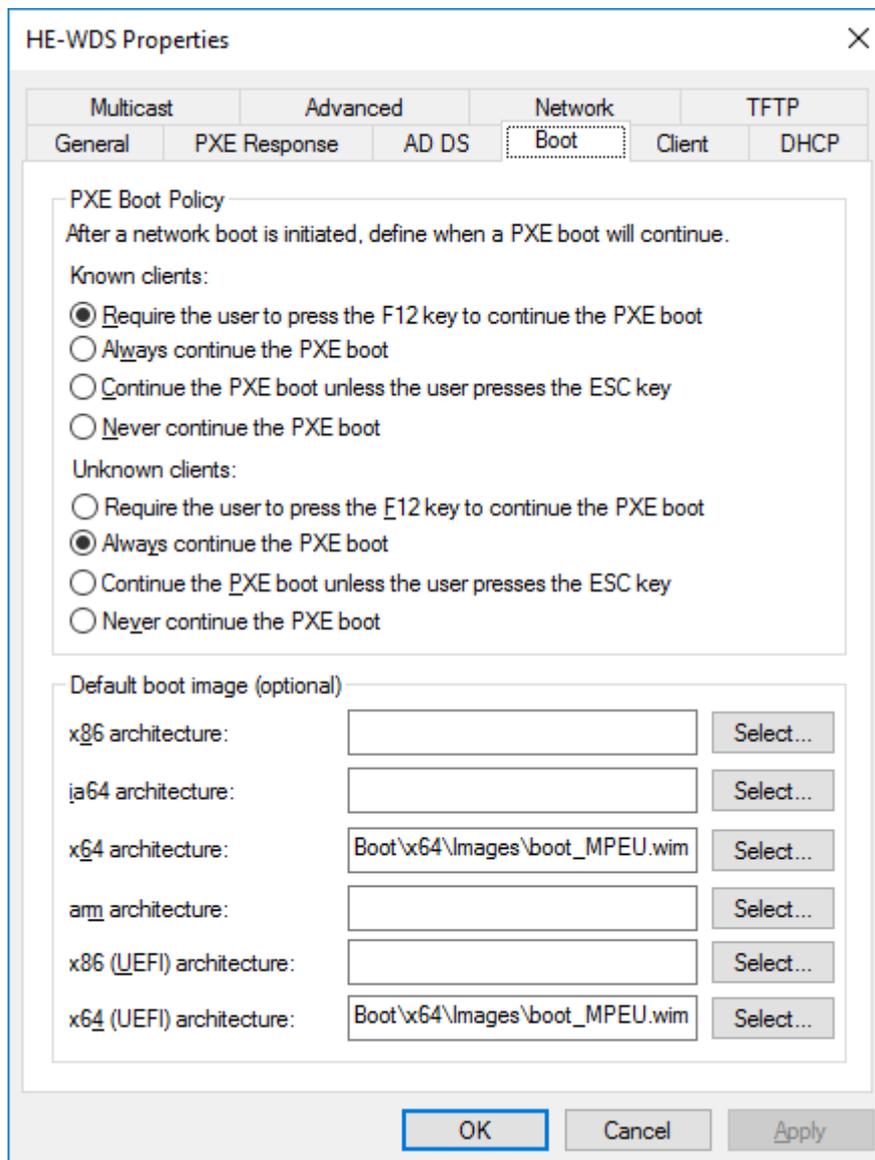


Figure 12: Default boot images

Figure 12 shows that the default image for each architecture can be selected individually. When selecting the boot image(s) for different architectures only the applicable images are offered to choose from. In this project, only the “x64 (UEFI) architecture” image is used for new installations, but the non-UEFI image is provided for legacy installations of Windows 7 operating systems. Here are also the options for the PXE boot policy, which should be chosen according to the company’s preference. The last option “Never Continue the PXE boot” can be used to disable the PXE boot.

4.5 Deploying Linux operating system

As explained in the theory section, the Linux deployment cannot be done as a standalone so instead the PXELINUX bootloader from the Syslinux project is used. The PXELINUX bootloader will be set as the default bootloader provided by the WDS server, but it also still allows to boot the Windows Boot Manager. In this study, the PXELINUX bootloader is only used for BIOS/Legacy boot, not for UEFI.

Here are the steps for adding the bootloader and other needed files to the existing WDS installation:

- Download the 4.04 SYSLINUX package from the official website
- Extract the following files: pxelinux.0, vesamenu.c32, chain.c32
- Rename the “pxelinux.0” -file to “pxelinux.com”
- Place these files to “RemoteInstall\Boot\%Architecture%”, under both x64 and x86 architectures
- Create folders “pxelinux.cfg” and “Linux” under both architectures
- Create a file named “default” without file extension under “pxelinux.cfg” -folder
This is used to configure the boot menu later
- Under both architectures make copy of “pxeboot.n12” and name it to “pxeboot.0”.
- Under both architectures make copy of “abortpxe.com” and name it to “abortpxe.0”.

The next step is to change the bootloader. Since the Windows Server 2008 R2, this can only be done using the command line tool “WDSUTIL” instead of using the GUI. The following commands set the pxelinux.com as the bootloader for both architectures:

```
wdsutil /set-server /bootprogram:boot\x86\pxelinux.com /architecture:x86
wdsutil /set-server /N12bootprogram:boot\x86\pxelinux.com /architecture:x86
wdsutil /set-server /bootprogram:boot\x64\pxelinux.com /architecture:x64
wdsutil /set-server /N12bootprogram:boot\x64\pxelinux.com /architecture:x64
```

Next the boot menu must be configured. The “default” file created earlier is used as a configuration file for the menu. The final menu schema is presented in figure 13:

```

DEFAULT      vesamenu.c32
PROMPT      0
NOESCAPE    0
ALLOPTIONS  0
# Timeout in units of 1/10 s
TIMEOUT     300
MENU TITLE  PXE Boot Menu(x64)
MENU INCLUDE pxelinux.cfg/graphics.conf
#---
LABEL wds
MENU LABEL  Windows Deployment Services
KERNEL     pxeboot.0
#---
LABEL Abort
MENU LABEL  AbortPXE
Kernel     abortpxe.0
#---
LABEL local
MENU DEFAULT
MENU LABEL  Boot from Hard Drive
LOCALBOOT  0
Type       0x80
#---
LABEL Ubuntu
MENU LABEL  Ubuntu 16.04.2
Kernel     /Linux/Ubuntu/casper/vmlinuz.efi
append     boot=casper netboot=nfs nfsroot=10.80.25.12:/Ubuntu initrd=Linux\Ubuntu\casper\initrd.lz
#---
LABEL Memtest
MENU LABEL  Memtest86+ 5.01
Kernel     /memtest/memtest
#---
LABEL System rescue cd
MENU LABEL  System Rescue CD 5.1.1
Kernel     /SystemRescueCD/isolinux/rescue64
append     initrd=SystemRescueCD/isolinux/initram.igz dodhcp nfsboot=10.80.25.12:/SystemRescueCD

```

Figure 13: Default configuration file

In figure 13, many of the files created earlier are used in the “Kernel” statements.

In this “default” file the boot menu is formed, and it also contains the information of which “kernel” or boot file the corresponding menu item will download. More information on what these options mean is presented in the next sections 4.5.1 and 4.6. [12]

4.5.1 Ubuntu Live CD

As mentioned in the theory part, the Ubuntu Linux installer contains a “Live CD” mode, which allows users to test the distribution before installing it to the hard drive. In this section, the Ubuntu image is added to the WDS server and setup with PXELINUX to allow it to be launched in the Live mode directly from the boot menu. At the time of writing this study the newest Ubuntu Long-Term Support (LTS) version is the 16.04.3, the LTS version is much more stable and better suited for enterprises. The 64-bit Ubuntu 16.04.3 installer (ISO) was downloaded from the official site.

The PXELINUX does support booting of ISO-files using a software called memdisk, but it does not really work with today's Linux distributions or with any modern software. Luckily, there are other options, but they require significantly more configuration. [18]

First the Ubuntu ISO-file is extracted under the “%RemoteInstall%\Boot\x64\Linux\” in its own folder, called simply “Ubuntu”. In this study, only the 64-bit version of Ubuntu is configured, but the steps would be the same for the 32-bit version. Ubuntu unfortunately does not support fetching all the needed files using TFTP directly and the performance of such download might also be poor. Instead, when the Ubuntu Live CD is chosen from the PXELINUX boot menu, only the kernel and initrd files are downloaded through TFTP to the client. After this, the Ubuntu needs the root of the filesystem. This will be provided by sharing the root directory using NFS.

The Windows Server 2016 supports NFS sharing but first the role must be installed using “Add roles and features wizard”. The name of the role and the location in the menu is presented in figure 14.

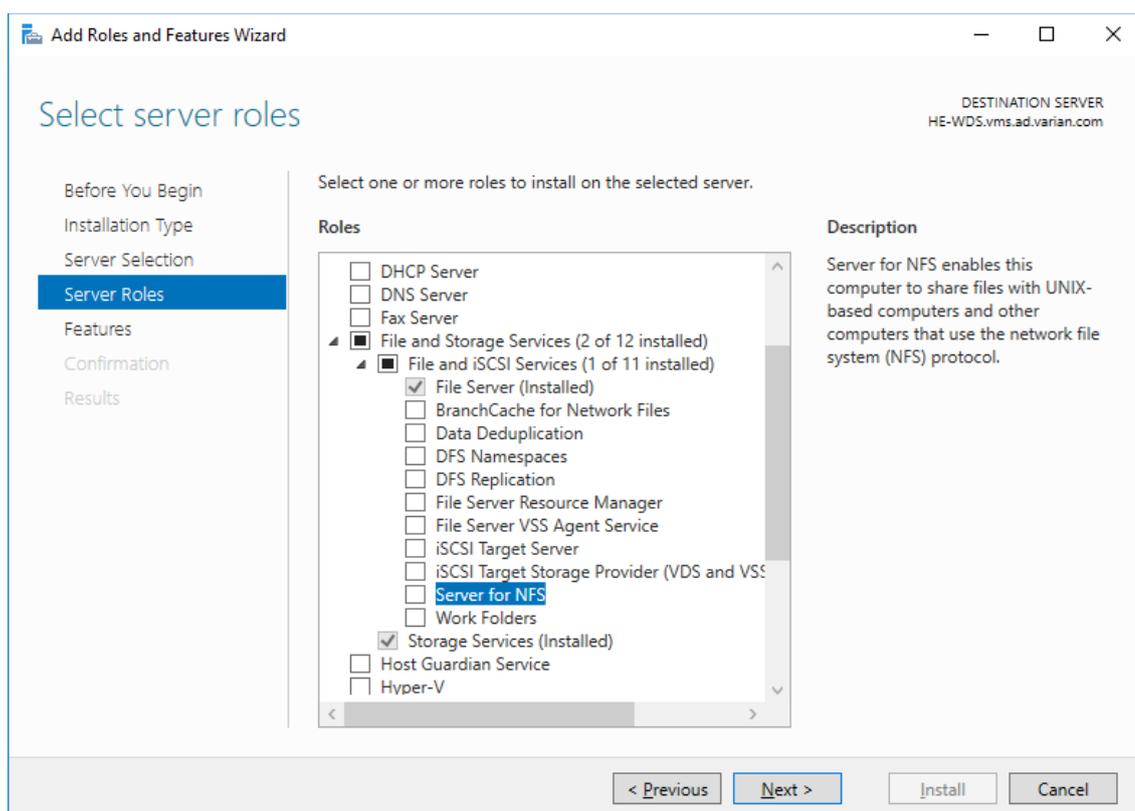


Figure 14: NFS Server role

As seen in figure 14, the “Server for NFS” option is located under the “File and Storage Services”. After the role is installed, the “Ubuntu” folder created earlier can be shared.

The NFS sharing can be enabled by right clicking the folder and opening the properties as presented in figure 15.

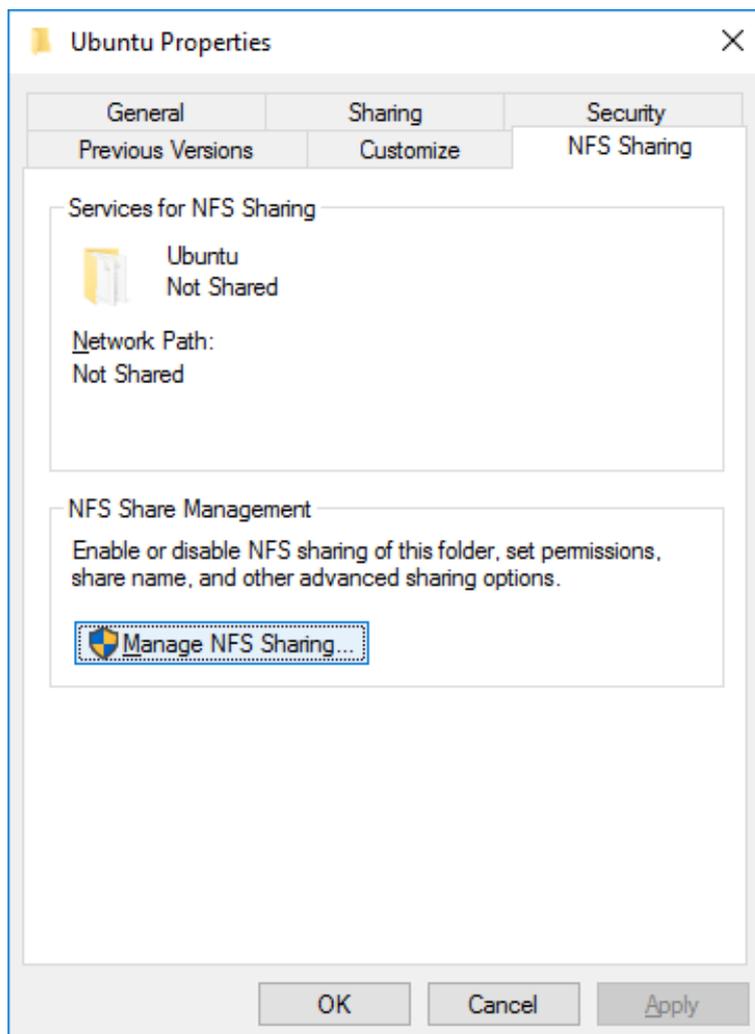


Figure 15: NFS Sharing

After the NFS sharing feature is added to the server, the “NFS Sharing” tab becomes available as seen in figure 15.

The default options can be used, just clicking the “Share this folder” check box and Apply completes the task. However, the permissions can be altered. The default options allow read-only access to “ALL MACHINES”, this is fine in the LAN environment of this study. Default configuration for the permissions is presented in figure 16:

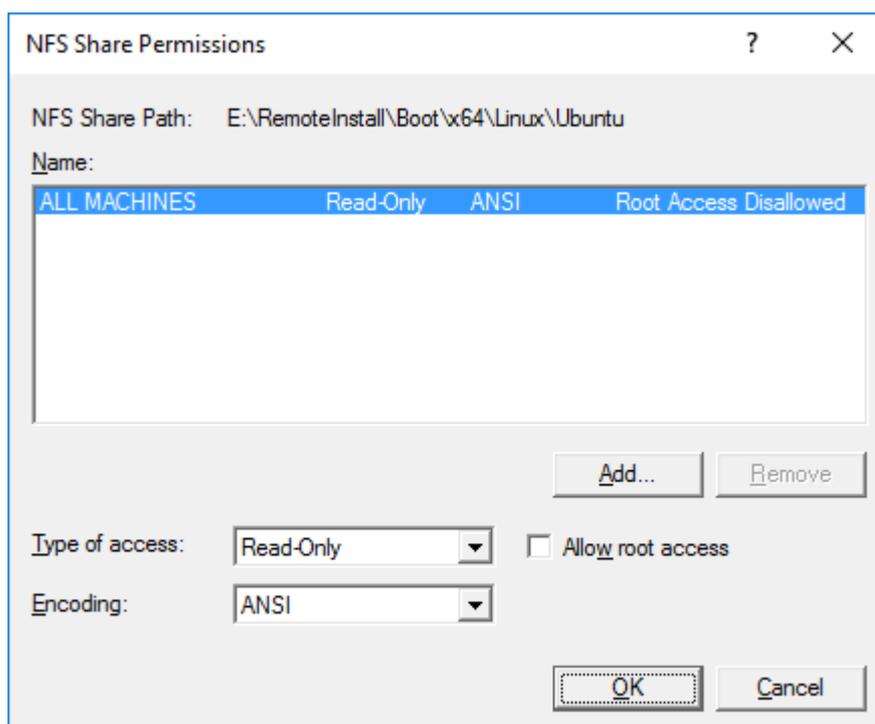


Figure 16: NFS permissions

The permissions seen in figure 16 allow the “Ubuntu” folder to be mounted by any client. It is possible to allow the NFS sharing for single IP addresses, but unfortunately adding subnets is not supported. Next, the PXELINUX boot menu must be configured with the right commands to load the kernel and initrd, and mount the root filesystem as seen in figure 13, the “default” file contains these lines:

```

LABEL Ubuntu
MENU LABEL Ubuntu 16.04.2
Kernel /Linux/Ubuntu/casper/vmlinuz.efi
append boot=casper netboot=nfs nfsroot=10.80.25.12:/Ubuntu initrd=Linux/Ubuntu/casper/initrd.lz

```

Going through this line by line, LABEL is just a note of what this entry is in the “default” file, now the next line MENU LABEL is presented in the boot menu GUI for the user. The next line is the Kernel which tells bootloader where the kernel file is and then the append line contains the rest of the kernel command.

The kernel commands are formed with the following logic:

`boot=casper` – Tells the kernel to boot into the Live CD mode, called casper in Ubuntu
`netboot=nfs` – Tells the kernel to netboot using NFS
`nfsroot=10.80.25.12:/Ubuntu` – Tells the nfsroot location to mount, IP address is for the NFS-server (same as the WDS server in this case) using a DNS name is not supported
`initrd=Linux\Ubuntu\casper\initrd.lz` – Tells where to find the initrd. Note that the root for the path is the same as for the kernel file

These commands are not universal, but they work with most Ubuntu-based distributions that support the Live CD/casper mode, for example Lubuntu. Even though Ubuntu is launched in the Live CD mode, it is possible to install the distribution to the hard drive using the installation wizard in the Live session just as when installing it from a CD/USB media. [12]

4.6 Adding backup and recovery tools

In this section, the backup and recovery tools presented in the theory chapter will be added to the WDS server and published using the PXELINUX boot menu. This section will show two different approaches to delivering these tools and tries to act as an example of how to publish similar tools using the WDS.

4.6.1 Memtest86+

Memtest86+ can be downloaded in many forms such as a pre-compiled ISO-file, binary file and even the source code can be downloaded as it is open source. For this setup with WDS the PXELINUX should support both the ISO and the binary file, but testing shows that the memdisk software method is not possible with the ISO file, thus it is best to use the binary file. The pre-compiled binary used is available from the official website. At the time of writing this thesis the newest software version is 5.01.

- After downloading the binary file, a new folder for Memtest86+ is created under “%RemoteInstall%/boot/x64/” called “memtest”.
- The binary file will be copied there and renamed to “memtest” without a file extension.

The only thing left to do is make a boot menu entry for Memtest86+ by editing the “default” file:

LABEL Memtest – General label for the menu entry

MENU LABEL Memtest86+ 5.01 – Label of the menu item visible in the GUI of the boot menu

Kernel /memtest/memtest – Location of the binary file, “%RemotInstall%/boot/x64/” as root

Figure 17 presents the GUI of Memtest86+:

```

Memtest86+ 5.01 | Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz
CLK: 3408 MHz (X64 Mode) | Pass %
L1 Cache: 32K 340752 MB/s | Test %
L2 Cache: 256K 113584 MB/s | Test #2 [Address test, own address Parallel]
L3 Cache: 8192K 262117 MB/s | Testing:
Memory : 2048M 170376 MB/s | Pattern: | Time: 0:00:00
-----
Core#: 0 (SMP: Disabled) | CPU Temp | RAM: 0 MHz (DDR3- 0) - BCLK: 0
State: W Running... | °C | Timings: CAS 19-15-15-31 @ 192-bit Mode
Cores: 32 Active / 1 Total (Run: All) | Pass: 0 Errors: 0
-----
S S
(ESC)exit (c)configuration (SP)scroll_lock (CR)scroll_unlock

```

Figure 17: VM running Memtest86+

The Memtest86+ binary file is only 147 KB, which is impressive as it still supports almost every x86 hardware. It also loads in a blink of an eye when selected from the boot menu. In figure 17, the GUI also shows the CPU temp, CPU model and memory information for the user.

4.6.2 System Rescue CD

As the “CD” in the name implies the System Rescue CD is a bootable software available as an ISO file and usually burned to a CD/DVD or saved to a USB storage device. Adding it to WDS is a bit more complex task than creating a removable media:

- First the ISO must be downloaded from the official website, at the time of writing this study the newest version is 5.1.1. Only the 64-bit version is configured in this example.
- New folder called “SystemRescueCD” is created under “%RemoteInstall%/boot/x64/”. The ISO file downloaded is extracted to the folder.

When inspecting the contents of the ISO, it is obvious that System Rescue CD is a Linux based system. Thus, the System Rescue CD is added to the WDS server in a fairly similar manner as the Ubuntu image.

The next step is to share the “SystemRescueCD” folder using NFS, and the steps are the same as for Ubuntu in the section 4.5.1. After the folder is shared, it is time to create the boot menu entry by editing the “default” configuration file:

```
LABEL System rescue cd
MENU LABEL System Rescue CD 5.1.1
Kernel /SystemRescueCD/isolinux/rescue64
append initrd=SystemRescueCD/isolinux/initram.igz dodhcp nfsboot=10.80.25.12:/SystemRescueCD
```

The first line LABEL is just a general label for this entry and the MENU LABEL is shown in the GUI of the boot menu. Next, the Kernel line is a path to the correct kernel for the architecture, in this case “rescue64” but for 32-bit systems it would be “rescue32”. In the append line is the rest of the kernel command, here the initrd path is given just as the “rescue64” kernel path. The next two entries are “dodhcp” which means that IP-address will be assigned by the DHCP server and nfsboot command points to the NFS-shared folder, which is used to access a file called “sysrcd.dat” which is the image of the root filesystem. NFS is not the only method to provide the “sysrcd.dat” file, NBD or HTTP could be used instead according to the documentation. The startup screen of System Rescue CD is illustrated in figure 18.

```

===== SystemRescue-Cd ----- 5.1.1 ===== tty1/6 ==
                        http://www.system-rescue-cd.org/

* Type net-setup eth0 to specify ethernet configuration.
* If your PC is on an ethernet local network, you can configure by hand:
  - ifconfig eth0 192.168.x.a (your static IP address)
  - route add default gw 192.168.x.b (IP address of the gateway)

* To be sure there is an ssh server running, type /etc/init.d/ssh start.
  You will need to create a user or to change the root password with passwd.

* Available console text editors : nano, vim, gemacs, zile, joe.
* Web browser in the console: elinks www.web-site.org.

* Ntfs-3g : If you need a full Read-Write NTFS access, use Ntfs-3g.
  Mount the disk: ntfs-3g /dev/sda1 /mnt/windows

* Graphical environment :
  Type startx to run the graphical environment
  X.Org comes with the XFCE environment and several graphical tools:
  - Partition manager:..gparted
  - Web browsers:.....firefox
  - Text editors:.....gvim and geany

root@sysresccd /root % _

```

Figure 18: System Rescue CD command line

As the figure 18 text gives away, the System Rescue CD also includes a set of GUI tools, but also a very powerful set of shell applications, and even popular programming languages Python, Perl and Ruby. [19]

4.7 Automating the boot image upload

To make updating the solution less complicated and time consuming a PowerShell script will handle most of the steps presented in the sections 4.3 and 4.4. Before the script is run, the system administrator still needs to import manually the new ISO file to the server. Some of the commands used by the script must be run as administrative account on the WDS server, thus the script will attempt to open a new PowerShell session as administrator if run without an elevated PowerShell session. Also, the script must be launched on the server using a remote PowerShell session with CredSSP authentication or a remote desktop session to connect to the WDS server. The script code is found in Appendix 1 and a flowchart of the program is presented below in figure 19.

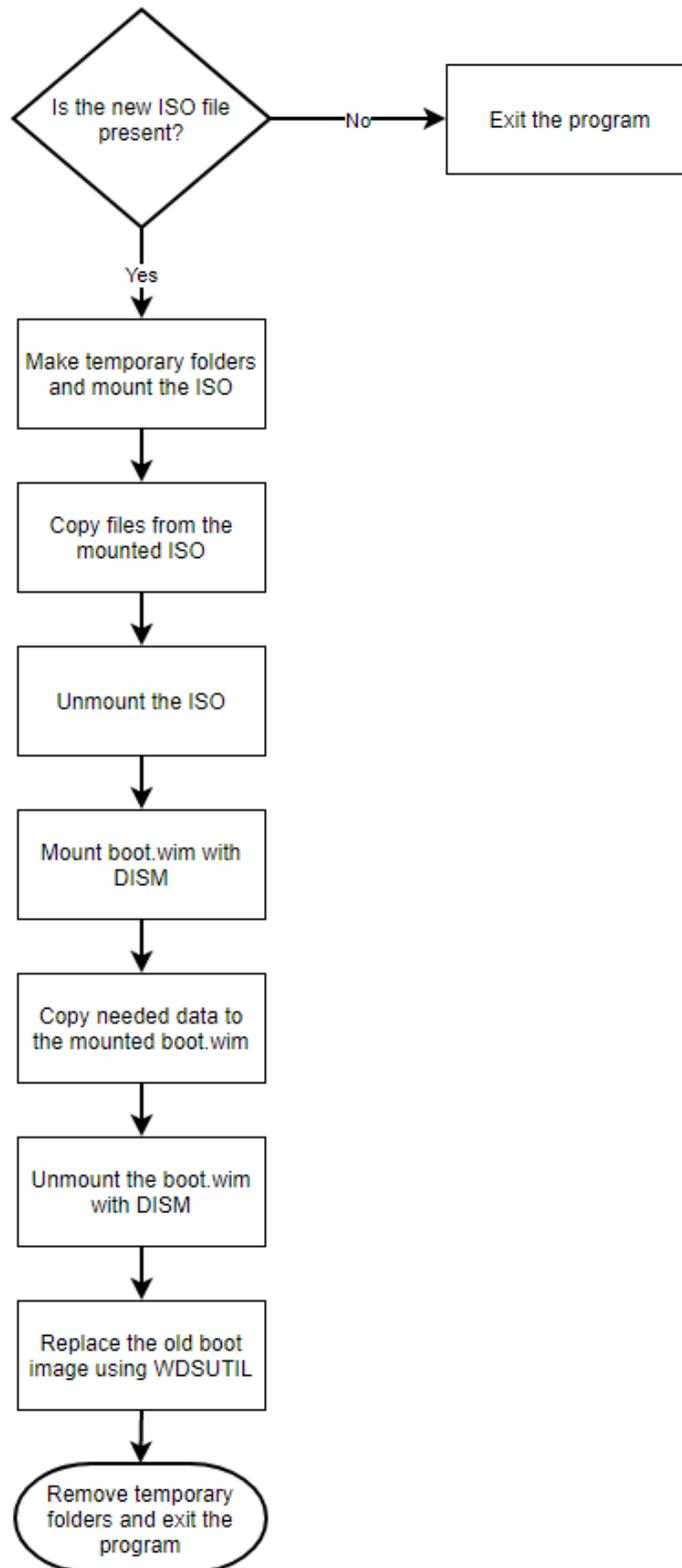


Figure 19: Flowchart of the PowerShell script

As the flowchart in figure 19 shows, the program does not interact with the user other than prompting the user to upload the image and then checks if an applicable file is found

before continuing. This completes the implementation and the next section will focus on testing and verifying the solution.

5 Testing and verification of solution

In this section, the solution presented in the previous section is tested and verified that it functions as intended. Theory and specifications present how the solution should work, but the testing will verify if the theory is correct and whether the systems work as they are described.

5.1 Network

To test the network, a software called Wireshark was used to capture and inspect the data transfer through the network. The client machine used for the testing was a VM hosted on a desktop computer with a virtual bridged NIC. Wireshark was also used on the server to verify that data transfer is successful, and no data units are dropped between the client and server.

5.1.1 UEFI PXE boot

Inspecting the packets verifies that the data transfer between client, DHCP server and WDS server happens in the following sequence:

1. Client sends a DHCP discover containing option 93 indicating that the architecture of client is "EFI x86-64 (9)" and option 60 indicating "PXEclient" to UDP port 67
2. DHCP server answers with a DHCP offer with the IP address for the client to UDP port 68
3. WDS server answers with a DHCP offer announcing its IP address to the client and Option 60 containing the identifier "PXEclient" to clients UDP port 68
4. Client sends a DHCP Request to accept the IP address provided by DHCP server to UDP port 67
5. DHCP server answers with DHCP ACK to client UDP port 68

Note: All DHCP frames to this point are sent as broadcast and network devices use IP helper to forward them to the server(s)

6. Client now has IP address and it notices that the WDS server is in another subnet, so it performs ARP to find out the MAC address of the default gateway
7. Client sends unicast packet to WDS server containing the same information as option 60 in the payload to UDP port 4011
8. WDS server answers with the boot file name: "boot\x64\wdsmgfw.efi" to UDP port 4011
9. Client sends TFTP read request for the boot file to UDP port 69
10. WDS server and Client exchange the boot file with TFTP using random UDP ports for example client used port 1435 and WDS server 57257, this concludes the PXE boot and after the download of the file it is executed
11. After the "wdsmgfw.efi" file is downloaded the client sends a DHCP request to WDS server UDP port 4011
12. WDS server answers with another boot file name "boot\x64\bootmgfw.efi" to UDP client port 68
13. After the "bootmgfw.efi" -file, BDC file, multiple policy files and a font file are downloaded the boot image is downloaded and executed

From the initial DHCP discover frame to the beginning of downloading the WIM file it takes ~35 seconds. It takes ~30 seconds from the first TFTP block to the acknowledgment of the last block of the 402 MB WIM file. The user is also provided with a download bar presented in figure 21:



Figure 2: Downloading the image

As figure 21 illustrates, the system administrator is provided with useful information of the process throughout the PXE sequence. Possible error codes would also be displayed on the screen. The same inspection was done also for BIOS/Legacy boot client and the results between UEFI and BIOS/Legacy are discussed in the next section.

5.2 Comparing UEFI and BIOS/Legacy PXE boot

Microsoft's WDS executes the PXE just as the PXE specification by Intel describes it. Because of this, the UEFI and BIOS/Legacy boot modes are almost identical. In the last section, the UEFI boot sequence was explained in detail, and these steps are the same for BIOS/Legacy boot with few differences. Table 1 shows the steps which differ between the UEFI and BIOS/Legacy boot sequence.

Step	UEFI	BIOS/Legacy
1	Option 93 "EFI x86-64 (9)"	Option 93 "IA x86 PC (0)"
7	General unicast packet to WDS server	DHCP discovery packet to WDS server
8	Boot file name "boot\x64\wdsmsgfw.efi"	Boot file name "boot\x86\wdsnbp.com"
n/a	Time to download boot.wim ~30 seconds	Time to download boot.wim ~14 seconds

Table 1: UEFI and BIOS comparison

The last steps that lead to downloading the boot image are different in BIOS/Legacy, because of the PXELINUX boot menu, thus it is marked as n/a in table 1. It was noticed that the TFTP download time of the boot image is considerably slower when using UEFI.

5.3 Hardware compatibility

The case company uses Dell as the primary vendor and all the current Dell models support Intel's PXE specification. Dell has also implemented support for the PXE boot through docking stations for the laptops. A variety of Dell desktop and laptop models were tested, and no faults or problematic models were discovered during the testing. This result is not a surprise as usually the problems in deployment happen during the installation and not in the initial network boot.

6 Security

Security is a big concern for organizations and there for the solution must be secured while keeping in mind the effectivity of the system. Not everyone should be able to use all the available features and images, thus a passphrase protection will be presented in this section.

6.1 Securing PXELINUX with a passphrase

The Syslinux package allows securing the whole menu with a master password or securing individual boot options with a password. To secure a boot option, the “default” menu configuration file must be edited to include the “MENU PASSWD” phrase after the “MENU LABEL” statement. Figure 22 presents the format:

```
# ---
LABEL wds
MENU LABEL Windows Deployment Services
MENU PASSWD $5$NVXVZSs73pcTwMIY$8TnHPUgKUPGGAYUB.xfqWkAt1amAru1uTT58DMZEmV9
KERNEL pxeboot.0
```

Figure 22: MENU PASSWD example

As seen in figure 22, the passwords can be encrypted. The Syslinux boot menu supports md5, sha1, sha256 and sha512 hashes. Compatible hashes can be created for example with Python’s “crypt” library. The hash shown in figure 22 was created with the following python command:

```
Import crypt; print(crypt.crypt("test", crypt.mksalt(crypt.METHOD_SHA256)))
```

The Python version used was 3.5.2. The hash generated by the command will be different from the one shown in figure 22 every time since the “mksalt” function provides a random 16-character cryptographic salt for the function. In figure 22, the salt is “NVXVZSs73pcTwMIY” located between the second and third \$-signs and the number “5” between first and second \$-signs means that the hash made with SHA-256 function.

7 Conclusions

For writing this study, an extensive research and studying of Windows Deployment Services, PXELINUX, PXE specification and PowerShell tools was conducted. Based on the research, the tools for the implementation were chosen and configured. The solution presented is straight forward and quick to configure and implement in any LAN environment. As the case company uses the Windows operating system primarily, the WDS was a logical starting point for the study, but this affected the Linux compatibility harshly as WDS does not support anything besides Windows operating systems and the WIM image format. Thankfully, the WDS could be modified with the PXELINUX to add the support for Linux and other bootable images.

The driver for the solution was to save the system administrators' time by making the deployment of operating systems, diagnosing issues with hardware and making backups of broken systems more effective. Also, the solution aims to replace all removable storage devices used for these purposes previously. Overall, the solution matches the specifications and the case company's needs accurately and delivers the service reliably. Utilizing the network to deliver boot images of all kinds is extremely useful and every organization benefits from it greatly.

7.1 Future improvements and ideas

As briefly mentioned in section 4.5, the PXELINUX bootloader was only used for the BIOS/Legacy boot. The reason for this is that the UEFI bootloader included in the newest Syslinux package quite simply does not function at all. A brief testing with the PXELINUX UEFI bootloader showed quickly that the Dell hardware is not compatible. Another reason is that the backup and diagnostic tools presented do not support UEFI boot, since the Windows deployment image is the only UEFI compatible system a UEFI menu would be pointless, but in future such menu would be needed if other UEFI compatible images would be added.

The script presented in Appendix 1, is currently using the WDSUTIL command line tool instead of using the PowerShell cmdlets, which were introduced in Windows Server 2012 R2. The reason for this is that the PowerShell cmdlets are poorly documented and lack

a direct function to replace images, which the WDSUTIL includes. In future, the PowerShell cmdlets might improve or replace the WDSUTIL and then the script should be updated to upload the image with the PowerShell cmdlet(s).

References

- 1 FOG Project (2016) *FOG Project* Available at: <https://wiki.fogproject.org/wiki/> (Accessed 31 Oct. 2017)
- 2 Masotta, P. (2017) *PXE Server for Windows (UEFI & BIOS) – Serva* Available at: <https://www.vercot.com/~serva/default.html> (Accessed 2-Oct-2017)
- 3 Masotta, P. (2017) *PXE/BINL - AN01: Windows Network Install* Available at: <https://www.vercot.com/~serva/an/WindowsPXE1.html> (Accessed 2-Oct-2017)
- 4 Aidan Finn, Darril Gibson and Kenneth van Surksun (2010) *Mastering Windows 7 Deployment*, John Wiley & Sons, Incorporated
- 5 Network Booting Windows, Microsoft, July 2008 <https://technet.microsoft.com/en-us/library/2008.07.desktopfiles.aspx> Visited 5-Jul-2017
- 6 Intel (1999) *Preboot Execution Environment (PXE) Specification*, Intel Corporation
- 7 Microsoft (2008) *Network Ports Used* Available at: [https://technet.microsoft.com/en-us/library/cc732918\(v=ws.10\).aspx](https://technet.microsoft.com/en-us/library/cc732918(v=ws.10).aspx) (Accessed 31-Oct-2017)
- 8 Msikma (2016) *Efi-simple* Available at: <https://commons.wikimedia.org/wiki/File%3AEfi-simple.svg> (Accessed 1-Sep-2017)
- 9 Andre Della Monica, Russ Rimmerman, Alessandro Cesarini, and Victor Silveira (2016) *Deploying Windows 10* Available at: https://blogs.msdn.microsoft.com/microsoft_press/2016/02/23/free-ebook-deploying-windows-10-automating-deployment-by-using-system-center-configuration-manager/ (Accessed 31-Oct-2017)
- 10 Cobbler Project (2017) *Cobbler* Available at: <http://cobbler.github.io/> (Accessed 31-Oct-2017)
- 11 Cobbler Project (2017) *About* Available at: <http://cobbler.github.io/about.html> (Accessed 31-Oct-2017)

- 12 Syslinux Project (2016) *WDSLINUX* Available at: <http://www.syslinux.org/wiki/index.php?title=WDSLINUX> (Accessed 31-Oct-2017)
- 13 Canard PC (2013) *Memtest86+* Available at: <http://www.memtest.org/> (Accessed 31-Oct-2017)
- 14 PassMark Software (2017) *MemTest86 History* Available at: <https://www.memtest86.com/history.htm> (Accessed 31-Oct-2017)
- 15 Francois Dupoux (2017) *System Rescue CD* Available at: <http://www.system-rescue-cd.org/> (Accessed 31-Oct-2017)
- 16 Microsoft (2015) *Windows Deployment Services Getting Started Guide for Windows Server 2012* Available at: [https://technet.microsoft.com/en-us/library/jj648426\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/jj648426(v=ws.11).aspx) (Accessed 19-Sep-2017)
- 17 Felipe Binotto (2012) *Import SCCM boot image to WDS* Available at: <http://fbinotto.blogspot.fi/2012/04/import-sccm-boot-image-to-wds.html> (Accessed 01-Aug-2017)
- 18 Syslinux Project (2017) *MEMDISK* Available at: <http://www.syslinux.org/wiki/index.php?title=MEMDISK> (Accessed 31-Oct-2017)
- 19 Francois Dupoux (2017) *PXE NETWORK BOOTING* Available at: http://www.system-rescue-cd.org/manual/PXE_network_booting/ (Accessed 31-Oct-2017)

PowerShell script for updating boot image

```

#elevate the powershell if needed
If (-NOT ([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole] "Administrator"))
{
    $arguments = "& '" + $myinvocation.mycommand.definition + "'"
    Start-Process powershell -Verb runAs -ArgumentList $arguments
    Break
}
# ask user to confirm if the iso is in place
$choice = Read-Host "Make sure you have the new .iso in the E:\
and it's the only iso file there! Type Y to continue, N to
abort"
Switch ($choice)
{
    Y {Write-host "Starting the program"}
    N {Write-Host "Exiting..."; exit}
    Default {"Wrong input, exiting..."; exit}
}
#go to the root
Set-Location E:\
# find the iso file
$isofile = Get-ChildItem *.iso
if($isofile -eq $null) {Write-Host "No iso file found, please
double check"; Start-Sleep -s 3; exit}
#make temp folders
mkdir SCCM
mkdir img
# mount the iso
$mountresult = Mount-DiskImage -ImagePath $isofile -Passthru
#now let's get the letter
$letter = ($mountresult | Get-Volume).DriveLetter
#copy files to the sccm folder
Start-Process -FilePath "robocopy.exe" -ArgumentList "{$letter}\
E:\SCCM /E" -Wait -NoNewWindow
#unmount the iso file
Dismount-DiskImage -ImagePath $isofile
#make sure the boot.wim is not read only
Set-ItemProperty -Path E:\SCCM\sources\boot.wim -Name IsReadOnly
-Value $false -Passthru
#move the iso file so that it's not interupting the program next
time
$isoname = $isofile.Name
Move-Item $isofile E:\old_iso\$isoname
#next up is the dism mount
dism /mount-wim /wimfile:E:\SCCM\sources\boot.wim /index:1
/mountdir:E:\img
#now copy files
Start-Process -FilePath "robocopy.exe" -ArgumentList
"E:\SCCM\SMS\DATA E:\img\SMS\DATA /E" -Wait -NoNewWindow

```

```
#unmount the wim
dism /unmount-wim /mountdir:E:\img /commit
#now the E:\SCCM\sources\boot.wim can be added to WDS service
WDSUTIL /Replace-Image /Image:"WinPE" /ImageType:Boot /Architec-
ture:x64 /ReplacementImage /ImageFile:"E:\SCCM\sources\boot.wim"
/Name:"WinPE"
# clean up after image is uploaded, this will remove the
boot.wim from E:\
Remove-Item E:\SCCM -Force -Recurse
Remove-Item E:\img -Force -Recurse
Read-Host "\n\nThe .iso file is placed into E:\old_iso in case
it's needed, press enter to exit the program"
```